

# SHFileOperation

Vulnerable to TOCTOU issues

Sean Barnum, Cigital, Inc. [vita<sup>1</sup>]

Copyright © 2007 Cigital, Inc.

2007-04-16

## Part "Original Cigital Coding Rule in XML"

Mime-type: text/xml, size: 7311 bytes

Attack Category	<ul style="list-style-type: none"><li>• Path spoofing or confusion problem</li></ul>		
Vulnerability Category	<ul style="list-style-type: none"><li>• Indeterminate File/Path</li><li>• TOCTOU - Time of Check, Time of Use</li></ul>		
Software Context	<ul style="list-style-type: none"><li>• File Management</li></ul>		
Location	<ul style="list-style-type: none"><li>• shlobj.h</li></ul>		
Description	<p>Copies, moves, renames, or deletes a file system object.</p> <p>SHFileOperation is vulnerable to TOCTOU attacks. This function is a use-category TOCTOU function. The file parameters in the struct parameter must be absolute paths or this function is not thread-safe.</p>		
APIs	Function Name	Comments	
	SHFileOperation	Need to examine SHFILEOPSTRUCT to find race parameter	
Method of Attack	<p>The key issue with respect to TOCTOU vulnerabilities is that programs make assumptions about atomicity of actions. It is assumed that checking the state or identity of a targeted resource followed by an action on that resource is all one action. In reality, there is a period of time between the check and the use that allows either an attacker to intentionally or another interleaved process or thread to unintentionally change the state of the targeted resource and yield unexpected and undesired results..</p> <p>Since this function relies on a *name* there is an opportunity for the attacker to substitute an undesired file in place between a check and this use.</p>		
Exception Criteria			
Solutions	Solution Applicability	Solution Description	Solution Efficacy
	Generally applicable.	Utilize a file descriptor	Effective

1. [http://buildsecurityin.us-cert.gov/bsi/about\\_us/authors/35-BSI.html](http://buildsecurityin.us-cert.gov/bsi/about_us/authors/35-BSI.html) (Barnum, Sean)

	version of stat/ fstat when checking.	
Generally applicable.	The most basic advice for TOCTOU vulnerabilities is to not perform a check before the use. This does not resolve the underlying issue of the execution of a function on a resource whose state and identity cannot be assured, but it does help to limit the false sense of security given by the check. Attempt to create the directory and then check status after the creation.	Does not resolve the underlying vulnerability but limits the false sense of security given by the check.
Generally applicable.	Limit the interleaving of operations on files from multiple processes.	Does not eliminate the underlying vulnerability but can help make it more difficult to exploit.
Generally applicable.	Limit the spread of time (cycles) between the check and use of a resource.	Does not eliminate the underlying vulnerability but can help make it more difficult to exploit.
Generally applicable.	Recheck the resource after the use call to verify that the action	Effective in some cases.

	was taken appropriately.
<b>Signature Details</b>	int SHFileOperation(LPSHFILEOPSTRUCT lpFileOp);
<b>Examples of Incorrect Code</b>	<pre> int main(int argc, char* argv[]) {     int r = del_dir("c:\\tmp\\ \\test","*"); }  int del_dir(char *path, char *ext) {     SHFILEOPSTRUCT so = {0};     char whole_path[MAX_PATH] = {0};     int res = -1;     int check_status;     struct stat statbuf;      check_status=stat(path, &amp;statbuf);     [...]      so.wFunc = FO_DELETE;     so.fFlags = FOF_NOCONFIRMATION;     so.fFlags  = FOF_NOERRORUI;     so.hwnd = NULL;     sprintf(whole_path,"%s\\%s \\0\\0",path,ext);     so.pFrom = (LPCSTR)whole_path;      /*remove files*/     res = SHFileOperation(&amp;so);      if(res == 0 &amp;&amp; !_stricmp(ext,"*"))     res = !RemoveDirectory(path);      return(res); } </pre>
<b>Examples of Corrected Code</b>	<pre> /* No check ... So no TOCTOU vulnerability */  /* Delete a folder that is not empty */  int main(int argc, char* argv[]) {     int r = del_dir("c:\\tmp\\ \\test","*"); }  int del_dir(char *path, char *ext) {     SHFILEOPSTRUCT so = {0};     char whole_path[MAX_PATH] = {0};     int res = -1;      so.wFunc = FO_DELETE;     so.fFlags = FOF_NOCONFIRMATION; </pre>

	<pre>so.fFlags  = FOF_NOERRORUI; so.hwnd = NULL; sprintf(whole_path, "%s\\%s \\0\\0", path, ext); so.pFrom = (LPCSTR)whole_path;  /*remove files*/ res = SHFileOperation(&amp;so);  if(res == 0 &amp;&amp; !_stricmp(ext, "*")) res = !RemoveDirectory(path);  return(res); }</pre>					
Source References	<ul style="list-style-type: none"><li>• <a href="http://msdn.microsoft.com/library/default.asp?url=/library/en-us/shellcc/platform/shell/reference/functions/shfileoperation.asp">http://msdn.microsoft.com/library/default.asp?url=/library/en-us/shellcc/platform/shell/reference/functions/shfileoperation.asp</a><sup>2</sup></li><li>• <a href="http://msdn.microsoft.com/library/default.asp?url=/library/en-us/shellcc/platform/shell/reference/functions/shfileoperation.asp">http://msdn.microsoft.com/library/default.asp?url=/library/en-us/shellcc/platform/shell/reference/functions/shfileoperation.asp</a><sup>3</sup></li><li>• <a href="http://www.codecomments.com/archive371-2005-6-529312.html">http://www.codecomments.com/archive371-2005-6-529312.html</a></li></ul>					
Recommended Resource						
Discriminant Set	<table><tr><td>Operating System</td><td><ul style="list-style-type: none"><li>• Windows</li></ul></td></tr><tr><td>Languages</td><td><ul style="list-style-type: none"><li>• C</li><li>• C++</li></ul></td></tr></table>	Operating System	<ul style="list-style-type: none"><li>• Windows</li></ul>	Languages	<ul style="list-style-type: none"><li>• C</li><li>• C++</li></ul>	
Operating System	<ul style="list-style-type: none"><li>• Windows</li></ul>					
Languages	<ul style="list-style-type: none"><li>• C</li><li>• C++</li></ul>					

## Cigital, Inc. Copyright

Copyright © Cigital, Inc. 2005-2007. Cigital retains copyrights to this material.

Permission to reproduce this document and to prepare derivative works from this document for internal use is granted, provided the copyright and “No Warranty” statements are included with all reproductions and derivative works.

For information regarding external or commercial use of copyrighted materials owned by Cigital, including information about “Fair Use,” contact Cigital at [copyright@cigital.com](mailto:copyright@cigital.com)<sup>1</sup>.

The Build Security In (BSI) portal is sponsored by the U.S. Department of Homeland Security (DHS), National Cyber Security Division. The Software Engineering Institute (SEI) develops and operates BSI. DHS funding supports the publishing of all site content.

1. <mailto:copyright@cigital.com>